Web Services: A Design Outline

Steven Bucksbaum, February 20, 2011

1) A Web Service:
   a) Example of code reuse on a business or enterprise level.
   b) Define each web service in business functional terms.
      i) The web service interface supports the business function.
   c) Contract between two independent programs (can be running on different computers)
      i) Contract is independent of hardware technology
      ii) Contract is independent of the programs technology
      iii) Once developed and implemented, contract should remain stable over time.
2) Web Service Interface Design Considerations
   a) Interface is the <u>façade pattern</u>, a gateway to a larger processing application.
   b) Security:
      i) http versus https versus key-store
      ii) The user needs a login and password.
   c) Complex Interface versus Simple Interface:
      i) A complex interface is an aid for security. But when a change is made in the interface, all users must be informed and then change their code to correspond.
      ii) A simple interface: takes an XML string and returns an XML string.
   d) XML: Allows for changes that will not break the web service clients.
3) Coding Design and Implementation:
   a) Use N-Tier system design. The web service interface is the client interface layer.
   b) Cache reused database data to improve response time.
   c) Error messages are assigned unique numbers so they can be quickly located.
   d) Errors are split into two groups
      i) Catastrophic errors or errors that mean 'failure' throw exceptions.
      ii) In a high security situation, all errors throw exceptions.
      iii) Non-failure errors return information of what is needed.
   e) Use reflection for increased flexibility.
   f) Software Patterns: use where possible.
4) Data Access:
   a) Data Dictionary: each attribute name has a specific and unique definition.
   b) When using stored procedures.
      i) No business logic in the stored procedures.
   c) SQL-Injection (security issue): create a database function that checks all S/P arguments for characters that may be an SQL-Injection attack.
5) Testing and The Test System:
   a) Test System: Needs a parallel system to allow both the web service publisher and the web service client to test their code.
   b) Develop a "Test Driven" system.
   c) Data Quality Testing: Does the client receive the proper data requested?
   d) Response Testing: Does the web service provide adequate response to a query?
   e) Regression Testing: Create test code that automatically runs tests and provides regression testing as the code changes.
   f) Load Testing: Can the web service handle the anticipated load?
   g) User Acceptance Testing: Need a system for users to test their connection and code.
6) Other Considerations:
   a) Customer Support: Write a customer support document.
   b) Provide technical support for customers connecting to the web service.
   c) Lesson Learned. Do not assume the technical abilities of your web service consumers, they will need assistance.